CLAIM AMENDMENTS

This listing of claims will replace all prior versions, and listings, of claims in the application:

1. (Currently Amended): A method of performing multiple operations on a memory device, comprising:

dividing the memory device into k partitions, wherein k is an integer greater than or equal to two;

performing code operations from m code partitions out of k total partitions, wherein m is an integer greater than or equal to one;

performing data operations from n data partitions out of k total partitions through functions accessed directly from the code partitions at approximately the same time as the code operations are performed from the m code partitions, wherein n is an integer greater than or equal to one; and

suspending the data operations of the n data partitions if at least one of the functions accessed directly from the code partitions determines that a preempting operation with priority is detected.

2. (Original): The method of claim 1, wherein the data partitions and the code partitions do not overlap each other in the memory device.

3. (Original): The method of claim 1, wherein the m code partitions and the n data partitions equal the k total partitions.

4. (Original): The method of claim 3, wherein each of the m code partitions are equal in size to each of the n data partitions.

5. (Original): The method of claim 3, wherein the m code partitions and the n data partitions are fixed in memory space.

6. (Original): The method of claim 1, wherein the memory device is a flash memory.

7. (Original): The method of claim 6, wherein the flash memory is a flash electrically erasable read only memory (EEPROM) array.

8. (Currently Amended): An apparatus comprising:

logic for partitioning a memory device into a first plurality of partitions for storing code and a second plurality of partitions for storing data to enable multiple operations to be performed on the memory device at the same time;

logic for setting each of the partitions to a status mode to track operations performed on the memory device; and

logic for determining if a first requested operation has priority over a second requested operation, wherein the [[means]]logic for determining is stored within the first plurality of partitions for storing code.

9. (Original): The apparatus of claim 8, further comprising a means for saving a preempted operation before entering an interrupt routine.

10. (Previously Presented): The apparatus of claim 8, further comprising a means for restoring a preempted operation following an interrupt routine.

11. (Previously Presented): A memory array, comprising:

a data partition;

a code partition;

a status mode to provide a partition status from the memory array if a task request is received by the data partition, wherein if the partition status is busy, an algorithm in the code partition determines whether the task request preempts an existing task;

a read mode to enable code and data to be read from the memory array; and

a write mode to enable data to be written to the memory array.

12. (Original): The memory array of claim 11, wherein the code is programmed into the memory array.

13. (Previously Presented): The memory array of claim 11, wherein the write mode enables erase operations to be performed on data stored in the memory array.

14. (Original): The memory array of claim 11, wherein the memory array is a flash memory array.

15. – 18. (Cancelled)

19. (Currently Amended): An apparatus, comprising;

a memory device having a code partition and a data partition, wherein the code partition includes a function that is performed on data stored in the data partition; and

a flag to indicate when a suspend operation has occurred, wherein the function determines that the suspend operation has occurred if a requested second task of the data partition has [[a]]priority [[than]] over a first task of the data partition.

20. (Cancelled):

21. (Original): The apparatus of claim 19, wherein the memory device is a flash memory.

22. – 24. (Cancelled)

25. (Previously Presented): A method, comprising:

running a first operation of a first partition of a flash memory array;

running a first operation of a second partition of the flash memory array;

requesting a second operation to be performed on the second partition; and

determining from the first operation of the first partition if the second operation of the second partition has a higher priority than the first operation of the second partition.

26. (Previously Presented): The method of claim 25, further comprising:

suspending the first operation of the second partition if the second operation has a higher priority than the first operation.

27. (Previously Presented): The method of claim 26, further comprising:

setting a flag to indicate that the first operation of the second partition must resume after the second operation is completed.

28. (Previously Presented): The method of claim 26, further comprising:

running the second operation of the second partition.

29. (Previously Presented): The method of claim 25, further comprising:

ignoring the request to perform the second operation of the second partition if the first operation has a higher priority than the second operation.

30. – 34. (Cancelled)

31. (New) The method of claim 1, wherein the functions accessed directly from the code partitions comprises the functions accessed from the code partitions without first copying the functions to another memory device.